**Programa Institucional de Bolsas de Iniciação Científica – PIBIC**

**RELATÓRIO PARCIAL**

IDENTIFICAÇÃO DO PROJETO

Título do Projeto:\_\_ Automação De Sistemas Elétricos De Potência Com Dispositivos Eletrônicos Inteligentes\_\_

Local de Realização (Unidade/Instituto/Departamento/Laboratório): Escola de Engenharia / Universidade Federal Fluminense / Departamento de Engenharia Elétrica / Laboratório de Eletrônica \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Endereço: Rua Passo da Pátria, 156 – Bloco E, Sala 420A\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Bairro: São Domingos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Cidade: Niterói\_\_\_\_\_\_\_\_\_\_\_ \_\_UF:\_RJ\_\_ \_CEP:24210-240\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Matrícula:**.\_\_\_\_\_\_\_.\_\_\_\_\_\_\_-\_\_\_\_ **C.R.:** \_\_\_\_\_\_\_\_

Curso: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DADOS DO ORIENTADOR

Nome: Rainer Zanghi\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Matrícula Siape:\_2393115\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_CPF:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Endereço: \_Rua Passo da Pátria, 156 – Bloco D, Sala 421\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Bairro:\_São\_Domingos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Cidade:\_Niterói\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_UF:\_RJ\_\_\_\_\_\_\_\_\_\_\_\_CEP:\_24210-240\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_E-mail:\_rzanghi@id.uff.br\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Telefone 1: (21\_\_) 2629-5368\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Telefone 2: (\_\_\_\_\_)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DADOS DO BOLSISTA

Nome: Lucas Abdalla Menezes\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Matrícula: 116.038.019\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_CPF:171.509.847-11\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_CR:6.9\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Curso/Departamento/Instituto: Engenharia Elétrica/ Departamento de Engenharia Elétrica – TEE/ Universidade Federal Fluminense - UFF \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Endereço: Rua Santa Rosa 91, Apto. 906\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Bairro:\_Santa\_Rosa\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Cidade:Niterói\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_UF:RJ\_\_\_\_\_\_\_\_\_\_\_\_\_\_CEP:24240225\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Email:lucasabdalla@id.uff.br\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Telefone1:(21)971475033\_\_\_\_\_\_\_\_\_

##### 

**1. TÍTULO**

AUTOMAÇÃO DE SISTEMAS ELÉTRICOS DE POTÊNCIA COM DISPOSITIVOS ELETRÔNICOS INTELIGENTES

**2. INTRODUÇÃO**

A integração dos sistemas de automação de redes elétricas de potência tem demanda crescente de dispositivos de controle e supervisão inteligentes. Algoritmos da inteligência computacional (IC) podem ser aplicados a diversas funções desempenhadas pelos sistemas de automação atuais, ainda com paradigma predominantemente especialista. A migração de conteúdos e conceitos da Computação para a Engenharia Elétrica foi intensificada na automação de sistemas de energia elétrica com a crescente adoção da norma IEC 61850. No entanto, ainda é necessário articular os dois campos conceituais na formação de profissionais aptos, sendo necessárias abordagens trans e interdisciplinares. O presente projeto tem como objetivo investigar a convergência entre software e hardware na implementação de dispositivos eletrônicos inteligentes na automação. Serão exploradas implementações de algoritmos de inteligência computacional em hardware dedicado à automação e investigadas abordagens para extração de informação proveniente da leitura de grandezas elétricas.

**Veja o projeto e aproveite algum texto dali. Use também o que você já escreveu para o artigo.**

**3. METODOLOGIA**

Etapa 1:

A primeira etapa do projeto foi destinada ao estudo de bibliografias recomendadas pelo orientador, sobre alguns aspectos iniciais que viriam a ser encontrados durante o desenvolvimento da pesquisa. Foram explorados os seguintes temas em artigos científicos, Teses e trabalhos de conclusão de curso de graduação: Técnicas para medição de corrente de maneira não intrusiva em sistemas de distribuição de energia elétrica e tipos de algoritmos da inteligência computacional para identificação de padrões. (Falta Indicar as Referências).

Etapa 2:

Nessa etapa, foi iniciado o desenvolvimento do artigo sobre o uso de sistemas embarcados de baixa custo para o monitoramento não intrusivo de cargas ou NILM (do inglês Non-Intrusive Load Monitoring). Além do estudo de outras bibliografias para investigar os requisitos mínimos de software e hardware do sistema para o tratamento dos dados, de maneira que a proposta de reconhecimento de cargas não fosse afetada.

Etapa 3:

Na terceira etapa, foram estudadas plataformas computacionais de baixo custo disponíveis no mercado e sua aderência aos requisitos mínimos de software e hardware escolhidos na etapa anterior. Foi feito também, o projeto do circuito para medições de corrente, incluindo a seleção de componentes para montagem do circuito.

Etapa 4:

Nessa etapa, foi feita a montagem de um protótipo do circuito projetado, e iniciada a elaboração de um código que possa ser reaproveitado para os demais dispositivos que serão testados. Esse código é responsável por fazer a aferição de corrente de uma malha do circuito, e então faz o cálculo do seu valor RMS para cada instante, dado um certo tempo de amostragem.

**4.RESULTADOS**

**4.1 Elaboração do ambiente de testes.**

Após estudados métodos de extração dos valores de corrente, o primeiro passo foi montar uma esquemática do ambiente de testes. Na Figura 1, está ilustrado um diagrama com os elementos utilizados no ambiente de testes.



Figura 1: Diagrama do ambiente de testes

O esquema funciona da seguinte forma: Uma corrente atravessa o TC, que por sua vez, produz uma tensão no seu enrolamento secundário, que é proporcional ao fluxo enlaçado gerado por essa corrente. Essa tensão está contida no intervalo de [-1,1] V. O micro controlador utilizado suporta apenas valores de tensão positivos entre 0 e +3,3V na sua porta de leitura para o conversor analógico -digital (ADC). Portanto, foi criado um circuito, que fosse capaz de produzir um deslocamento do nível de tensão CC (Offset), que garanta que essa tensão seja estritamente positiva, e dentro destes limites. Nesse caso, considerando a maior variação possível do sensor, ficará entre [+0,65 e +2,65] V. Depois disso, foi dado início ao desenvolvimento de um código que fosse capaz de ler a tensão na porta do ESP32 (porta 34, no nosso caso), e que fizesse a conversão desse valor para equivalência em Amperes RMS que atravessa o TC. A descrição dos três elementos será feita nas seções a seguir.

**4.1.1Um sensor de corrente não-intrusivo - transformador de corrente (TC) tipo janela.**



*Figura 2: TC tipo janela não intrusivo*

O esquema de funcionamento é descrito por Figura 3.

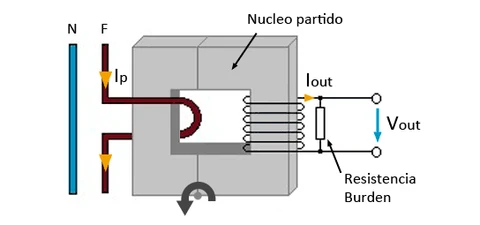


Figura 3: Esquema de funcionamento do TC

|  |  |
| --- | --- |
| O TC, pode ser descrito por (1) | 1 |

Onde,

- Ir é a corrente produzida no enrolamento secundário;

- Ip é a corrente correspondente ao número de condutores que atravessam o núcleo magnético do TC;

-VP é a tensão no terminal primário;

-VS é a tensão produzida na Resistência Burden;

-NP /Ns , representa a relação de transformação;

A tensão de saída fornecida pela TC, que é proporcional à corrente que o atravessa, é entregue por meio de um conector Jack 3,5 mm, que é muito comum no campo de áudio [6]. Onde a tensão de saída é dada pela diferença de potencial entregue pelos contatos, “L” e “K” do conector, representada pela Figura 4.

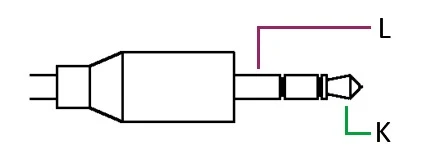


Figura 4: Conector Jack 3,5 mm

**4.1.2 Circuito de condicionamento do sinal do sensor – deslocamento de nível CC (Corrente Contínua).**

Uma vez que a maioria dos microcontroladores só suporta tensão CC na sua entrada, para a criação do circuito, é necessário saber qual a corrente máxima que o TC é capaz de suportar sem que a sua saturação magnética seja atingida, a fim de tornar o sinal de saída estritamente positivo. Com os dados nominais do TC, temos que para uma corrente máxima de 30A, esse produz uma tensão de 1V. Dessa forma, foi montado um circuito que gera, pelo menos, 1Volt de Offset do sinal de entrada, já que essa seria a tensão máxima que o TC consegue reproduzir.

Seguindo [7], foi montado o circuito da Figura 5.



Figura 5: Esquema do circuito para o deslocamento CC do Sinal

O circuito foi alimentado com uma tensão de (3,3 V c.c.) para um offset de 1,65V (3,3 V /2). A tensão de saída V\_tc é exatamente a tensão produzida pelo secundário do TC, deslocada de 1,65 V positivamente. Essa portanto, assume valores de (1,65 – 1) V até (1,65V +1) V. Definindo o intervalo [0,65 V, 2,65 V] dependendo do valor instantâneo de corrente demandado pela carga.

Lista de componentes utilizados:

- 1 Trimpot de 500kOhms, separada igualmente entre R1 e R2;

- 1 Capacitor para filtrar sinais de alta frequência que chegam no circuito;

- 1 Conector p2 para receber o sinal de saída do TC;

- 1 Fonte CC de 3,3 V proveniente da própria placa de desenvolvimento ESP32 DEVKIT V4;

O circuito projetado foi montado em um protótipo, utilizando uma matriz de contatos ou protoboard. A Figura 6 ilustra o protótipo do circuito de condicionamento de sinais montado.

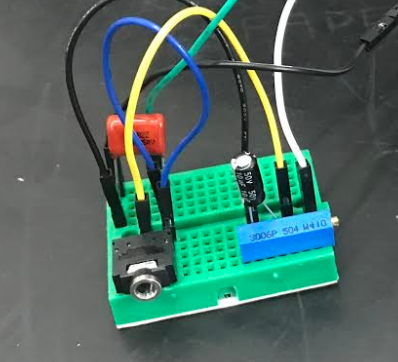


Figura 6: Circuito para o deslocamento CC do Sinal

**- 4.1.3 Uma plataforma computacional de baixo custo que será responsável por receber e tratar o sinal de saída do circuito montado.**

Para início da bancada de testes, foi utilizado um ESP32 [8]. pelas suas características um pouco mais robustas que outros sistemas computacionais de baixo custo na mesma faixa de preço (como Arduino). Para as principais diferenças entre os dispositivos, tendo como base o Datasheet dos fabricantes [8] e [9], apresenta-se uma comparação entre as características mais relevantes na Tabela 1.

Tabela 1: Comparação entre Arduino - ESP32

|  |  |  |
| --- | --- | --- |
| Descrição | Arduino UNO | ESP32 |
| Pinos de I/O | 23 pinos com 6PWM | 34 pinos com 16 PWM |
| Memória RAM/SRAM | 2 kB | 520 kB |
| Temporizadores | 3 timers de 16-bits e dois de 8bits | 4 timers de 64-bits |
| Memória Flash | 32 kB | 4 MB |
| Memória ROM/EEPROM | 4 kB | 448 kB |
| Wifi | Não possui | Possui |
| Frequência de Operaçao | 0 – 16 MHz | 80 – 240 MHz |
| Preço (estimativa em fev. 2020) | R$70,00 | R$40,00 |

Uma foto do dispositivo utilizado, montado em uma placa de desenvolvimento ESP32 DEVKIT V4 [9] se encontra na Figura 7.



Figura 7: ESP32 DEVKIT V4

Na Figura 8, foram juntadas as três partes abordadas anteriormente, e postas num diagrama.



Figura 8: Diagrama do Ambiente de testes completo

Na Figura 9, é mostrada a montagem completa do diagrama descrito na Figura 8.

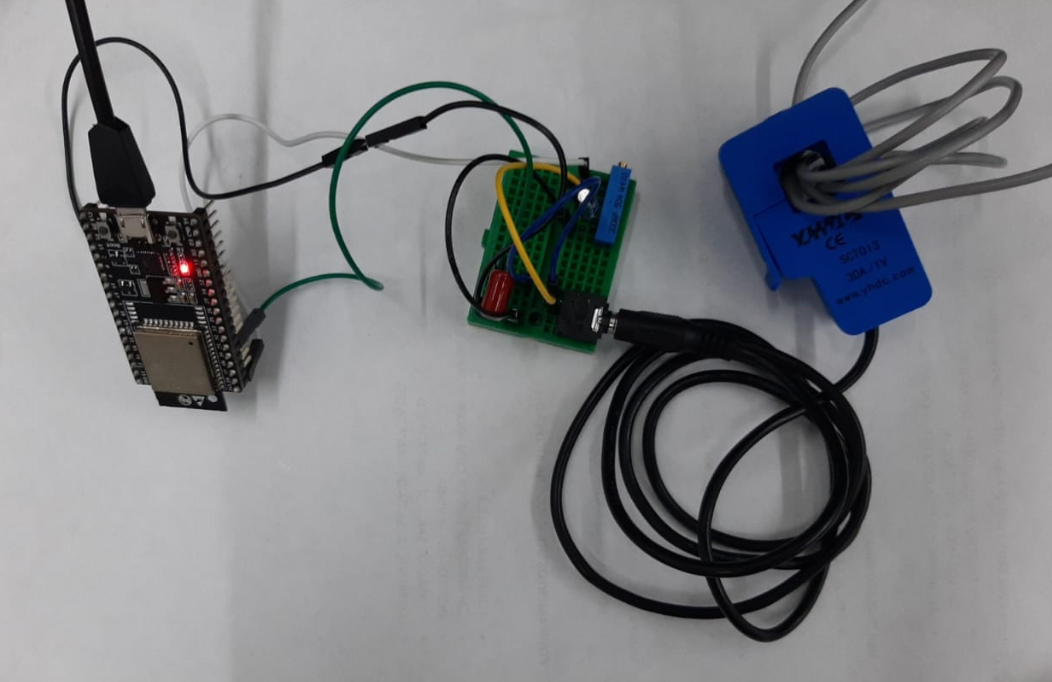


Figura 9 -Montagem do protótipo

**4.2 Produção de um código para o ESP32 que faça a leitura da corrente que passa pelo TC.**

**4.2.1 Idealização do Código**

Começando o desenvolvimento do código, o desafio se dá em identificar o quão rápido pode ser a frequência de amostragem, uma vez que o código seja escrito da forma mais otimizada possível.

O ESP32, consegue receber códigos de *Micro-Python* ou *C/C++,* como os demais microcontroladores do mercado. Utilizando Micro-Python, foi verificado que esse era muito inconstante e pouco confiável para tempos exatos de amostragem. Um simples código como o seguinte, gerava valores muitos distintos para cada valor da variável TEMPO\_DA\_ITERACAO, variando de . Isso ocorre, por falta de otimização na conversão de Micro-Python para C.

***VALORES\_LIDOS = [];***

***TEMPO\_PERCORRIDO = 0;***

***TEMPO\_DA\_ITERACAO;***

***PARA I de 0 ATÉ 255 FAÇA PASSO 1***

***VALORES\_LIDOS[I] = LER\_PORTA\_34();***

***TEMPO\_DA\_ITERACAO = TEMPO\_ATUAL – TEMPO\_PERCORRIDO;***

***ESCREVER(TEMPO\_DA\_ITERACAO);***

***TEMPO\_PERCORRIDO = TEMPO\_ATUAL;***

***FIM PARA;***

O mesmo código anterior escrito em C, ainda que gerando grandes variações, mostrou-se mais eficiente, sempre entregando valores dentro do intervalo de . Para a identificação das cargas, os intervalos entre amostras devem quase que constantes, para que esses não interfiram no resultado. Dessa forma, foi necessário a utilização de outra ferramenta para aquisição dos dados, além de uma simples iteração usando FOR.

Para isso, foi utilizado um timer/cronômetro do próprio ESP32, o uso de Interrupção e Região Crítica do código. O timer é responsável simplesmente por armazenar o tempo decorrido desde que o dispositivo foi ligado. A interrupção faz como que o processador pause a sua tarefa atual, e a atenda o dispositivo que chamou a interrupção. Já a Região Crítica de um código, garante que apenas aquele trecho de código, seja executada naquele instante, tornando a operação sobre recurso compartilhado atômica.

Então, esse código funcionaria da seguinte forma: O ESP32 é ligado, o timer é disparado, e uma função de Interrupção é associada ao timer, para que a cada essa função seja chamada. Quando essa é chamada, o código entra numa Região Crítica, e garante que apenas a leitura da porta digital esteja sendo executada naquele instante, otimizando o processo de leitura. Depois, que alguns valores fossem armazenados, uma função para o cálculo RMS dos valores obtidos seria chamada, e o timer reiniciado para recomeçar o processo. Além disso, o último valor RMS calculado fica disponível num servidor web, hospedado no próprio ESP32, e que pode ser acessado por protocolo HTTP.

Além disso, os valores gerados pelo conversor A/D do ESP32, variam de 0 a 4095, e esses, correspondem à uma tensão CC que também varia de 0 a 3,3 V. Dessa forma o nosso código, deve fazer essa conversão do valor em contagens fornecido pelo conversor A/D para Amperes. Utilizando a equação (1), para o cálculo da corrente instantânea que atravessa o TC, e a relação do transdutor de 0-30 A para 0-1 V temos:

|  |  |
| --- | --- |
|  | 2 |

- Counts: É o valor lido pela entrada analógica e convertido pelo conversor A/D;

-Counts\_ad\_cc: É o valor convertido pelo conversor A/D para a corrente de entrada do TC em 0;

- I\_máx\_tc: É o valor máximo de corrente medido pelo TC, especificado por [6];

- n: Constante que representa o número de condutores no interior do TC. Este recurso deve ser utilizado somente nos casos onde a corrente máxima medida é muito menor que o valor máximo de corrente do TC;

-V\_offset\_cc: É o valor de Tensão fornecido pelo circuito de deslocamento de nível de tensão.

**4.2.2 Produção do código**

A fim de otimizar e permitir o bom funcionamento do código, os seguintes artifícios foram usados:

- Definir constantes com ***#define***, para reservar um espaço em memória que não precisa ser acessado.

- Declaração dos timers que serão utilizados. O timer0 é responsável por pegar as amostras a cada 100 us e armazená-las num vetor de 256 espaços. O timer1 é responsável por resetar o timer0, a cada 500 ms.

- Declaração das variáveis com o modificador ***volatile,*** para evitar otimizações indesejadas do compilador. E uso de **DRAM\_ATTR** para garantir que a variável vá para a RAM que tem acesso mais rápido que a Flash.

- Declaração de um semáforo que identifica se no momento está ocorrendo um evento de interrupção. Uso de **uint32\_t**, para garantir 32 bits de memória para a variável.

- Associação de uma função ao **timer0** que é chamada a cada 100 us. Associando uma função ao **timer1** para disparar o **timer0**, que é chamada a cada 500 ms (O esquema fica como demonstrado na Figura 10).

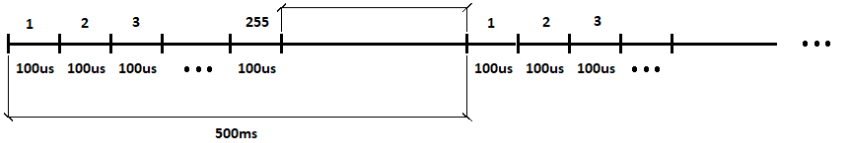


Figura 10: Funcionamento dos Timers para a leitura do sinal

- Criação de uma função que é chamada pela Interrupção, e entra em uma Região Crítica para fazer a leitura da porta 34 do controlador.

- Criação de uma função que é chamada pela Interrupção, a cada 500 ms para resetar o **timer0**.

- Função que retorna o último valor RMS calculado, ao acessar <https://IP.DO.DISPOSITIIVO/i_rms>.

Em termos de pseudocódigo, temos:

**importar** <WebServe**r>;**

**declarar** timer0, timer1: hw\_timer\_t ;

**declarar** ADC\_SAMPLES, I\_RMS\_VEC\_SIZE: INT;

**declarar** isrCounter: uint32;

**declarar** i\_rms\_data: float;

**declarar** adc\_data[]: uint32;

**...**

//função chamada pelo timer 0

**funcao onTimer**

Entrada\_na\_zona\_critica();

adc\_data[isrCounter] = leitura\_porta\_analogica(34);

isrCounter++;

Saida\_da\_zona\_critica();

**fim funcao**

**funcao onTimer1**

resetarTimer(timer0);

**fim funcao**

**funcao calcularRms : float**

**...**

**retorna valorRms;**

**fim funcao**

//Primeiro método chamado

**Setup**

Conectar.Wifi(login, senha);

Definir.endpoints();

AssociarFucoesAosEndpoints();

Iniciar.servidor();

Iniciar\_timer0();

Associar\_leitura\_a\_interrupcao\_do\_timer0();

Associar\_chamada\_do\_timer0\_ao\_timer1();

**fim Setup**

**funcao handleAcessoI\_rms**

i\_rms = calcularRms(adc\_data[]);

Servidor.enviar(i\_rms);

**fim funcao**

//chamada após o Setup até o desligamento do despositivo

**funcao Loop**

**verificar (timer\_ja\_disparado())**

**se** isrCounter++;

**verificar (timer0)**

**se**

parar\_timer\_0();

entrar\_em\_zona\_critica();

isrCounter = 0;

**para (int** i = 0; i < numero\_maximo\_de\_amostras; i++**)**

buffer\_adc\_data[i] = adc\_data[i];

**fim para**

sair\_da\_zona\_critica();

i\_rms\_data[i\_rms\_data\_idx] = i\_rms;

i\_rms\_data\_idx++;

**se** (i\_rms data\_idx >= I\_RMS\_VEC\_SIZE)

i\_rms\_data\_idx=0;

**fim verificar**

verificar\_requisicoes\_pro\_servidor();

**fim Loop**

Após inserido o código compilado no ESP32 , ao acessar <http://IP>, obtemos a última janela de valores completa usada para o cálculo do valor RMS. Acessando <http://IP/i_rms>, é obtido o valor RMS, para uma janela de intervalo de . E esse valor é atualizado a cada . Além disso, utilizando o endpoint <http://IP/i_rms_data>, é possível obter os últimos 512 valores, RMS calculados.

Foi produzido então, um código em OCTAVE, por conta da praticidade, para uma primeira visualização dos gráficos, e confirmar se as curvas obtidas condizem com a variação das cargas do circuito.

Para testar o bom funcionamento, o TC foi colocado em uma das fases no quadro geral, monitorando a corrente das cargas ligadas a ele.

Acessando <http://IP>, foram passados para um gráfico os valores obtidos, ficando como mostrado na Figura 12. Os valores presentes no eixo y representam as contagens fornecidas pelo conversor A/D para uma amostra realizada em laboratório e são mostradas na Figura 12 apenas para fins de ilustração.

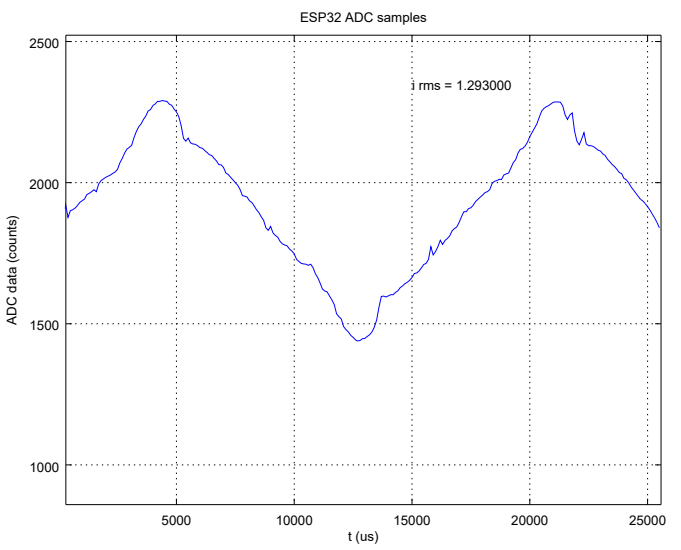


Figura 12: Exemplo de Curva de Corrente

Para testar a mudança de valores RMS com a variação das cargas, foram testados um micro-ondas e uma máquina de lavar.

Primeiro, o endereço <http://IP/i_rms_data> foi acessado, com as cargas que serão monitoradas desligadas como na Figura 13.

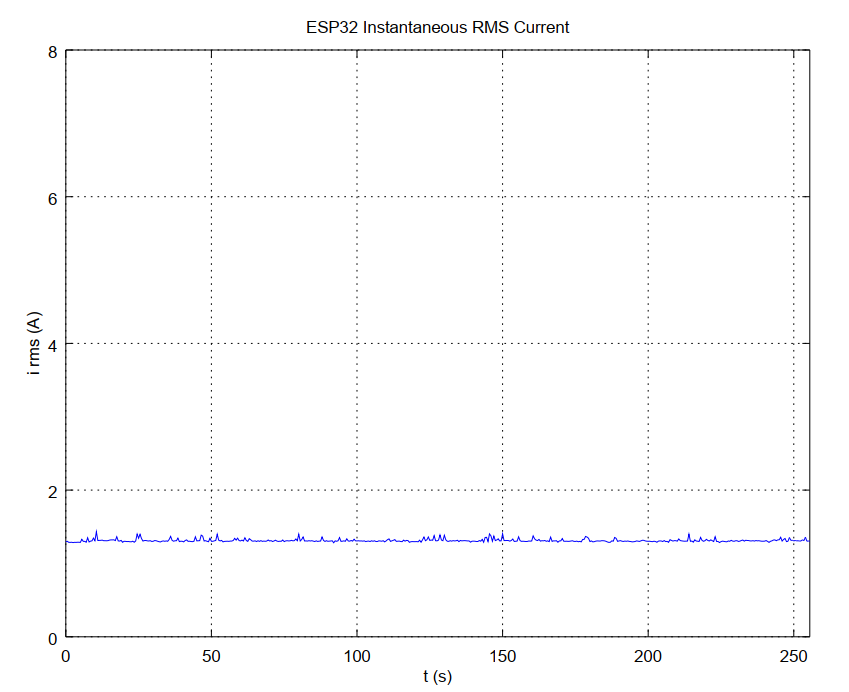


Figura 13: RMS no tempo para Micro-ondas e Máquina de lavar desligados

Logo em seguida foi iniciada a rotina de eventos de teste que teria as seguintes etapas e o tempo em que elas ocorreram:

1. Ligando a Máquina de lavar com o Micro-ondas desligado – t = 0 s;
2. Ligando a Micro-ondas – t = 60 s;
3. Desligando e Ligando o Micro-ondas rapidamente – t = 160 s;
4. Desligando e Ligando o Micro-ondas rapidamente – t = 240 s ;

Os resultados encontrados estão na Figura 14.

Os.

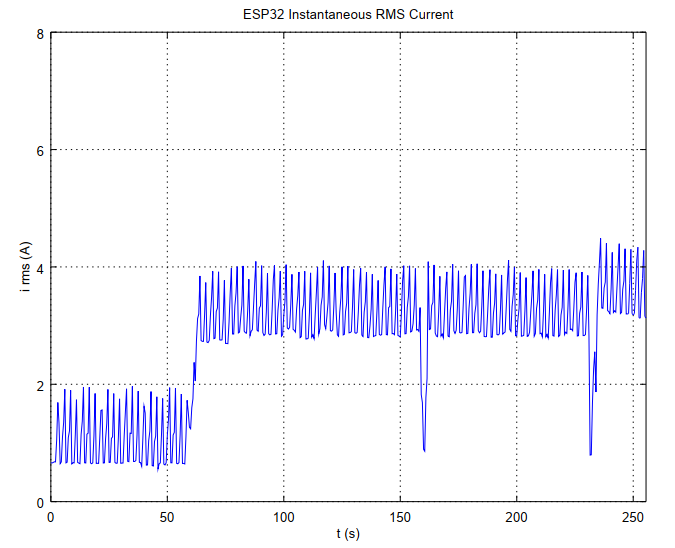


Figura 14: Parte 1 rotina de testes

Continuando os Testes anteriores:

1. Desligando e Ligando o Micro-ondas – t = 50 s;
2. Desligando e Ligando o Micro-ondas – t = 120 s;
3. Desligando o Micro-ondas – t = 170 s;
4. Desligando a Máquina de Lavar – t = 190 s;

Os resultados encontram-se na Figura 15 .

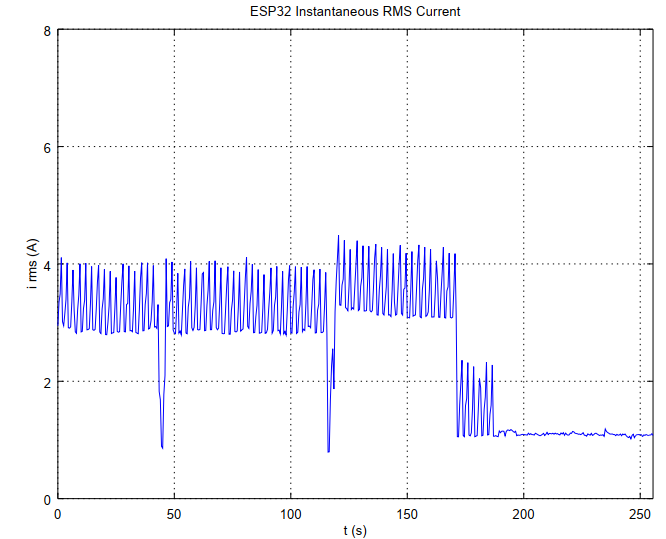


Figura 15: Parte 2 da rotina de testes

**5. PRODUÇÃO TÉCNICO-CIENTÍFICA**

Como citado na segunda etapa da parte de Metodologia desse documento, está sendo redigido um Artigo Científico, a partir de pesquisas feitas em base de dados como IEEE Xplore [10]. Esse artigo tem como intuito testar o funcionamento do monitorador de carga que está sendo produzido, e a partir dos resultados encontrados, sugerir futuras aplicações da técnica.

**6. CONCLUSÕES**

O objetivo da primeira parte do projeto, era de levantamento bibliográfico sobre métodos de medição de corrente não intrusivos, levantamento das possíveis plataformas disponíveis no mercado, leitura de sinais em micro controladores, e métodos para o tratamento de dados, considerando as limitações técnicas dos dispositivos.

Para a medição de corrente foi necessária a criação de um circuito para a leitura dos dados pelo ESP32. Logo no início da elaboração do código que seria inserido no dispositivo, foram encontradas limitações e dificuldades de processamento e acesso à memória, que dispositivos maiores, como computadores de mesa, não teriam.

Encontrados esses problemas, foi necessário ser feito mais estudos, sobre técnicas que otimizassem o processo de leitura, tornando-a eficiente e confiável. Após a conclusão da montagem do circuito para o deslocamento de sinal CC, que viria do TC, e finalização do código para o dispositivo, foi criado um pequeno script em OCTAVE, para visualização dos gráficos de corrente, e a variação seu valor RMS no tempo. Concluídas essas etapas, é esperado que o sinal que chega ao dispositivo seja confiável.

Nessa última parte, serão utilizadas as abordagens de identificação de sinais, estudadas no início do projeto, para a implementação de um modelo computacional, que seja capaz de classificar as cargas ligadas ao sistema, bem como diferenciá-las entre si.

**7. REFERÊNCIAS BIBLIOGRÁFICAS**

[6] Autor, “Título”, disponível em: <https://www.poweruc.pl/blogs/news/non-invasive-sensor-yhdc-sct013-000-ct-used-with-arduino-sct-013> . (Acessado em 07 de fevereiro de 2020)

[7] <https://learn.openenergymonitor.org/electricity-monitoring/ct-sensors/interface-with-arduino>

[8] <https://www.espressif.com/en/products/hardware/esp32/overview>

[9] <https://www.farnell.com/datasheets/1682209.pdf>

[10] <https://ieeexplore.ieee.org/Xplore/home.jsp>

**8. AUTO-AVALIAÇÃO DO ALUNO**

Com o aumento da relevância de sistemas de automação de redes elétricas de potências, a demanda de dispositivos responsáveis pela supervisão inteligente da rede, também aumentou. É interessante participar de um projeto que estuda a aplicabilidade de micro controladores, para o monitoramento da rede. Esse projeto traz para o meu contexto de graduando, a usabilidade de ferramentas computacionais para solução de problemas de Engenharia Elétrica. Foi possível também entender melhor conceitos de medição, funcionamento de circuitos integrados, e a partir da etapa atual do projeto, o uso de modelos computacionais focados em classificação. Além disso, o projeto está sendo importante para mostrar o funcionamento de metodologias científicas e como deve ser feito o planejamento para a realização da pesquisa. Por fim, pude ter contato com alguns temas que são vistos muito rápido durante o curso de graduação em Engenharia Elétrica, e aplicar alguns dos conceitos que já foram aprendidos nas cadeiras cursadas.